Lehrstuhl Informatik 4
Andreas Blume, Egor Kudrjaschow
Nils Dunker, Bastian Korte, Minsu So

technische universität dortmund

Wintersemester 2020/2021

Rechnernetze und verteilte Systeme
Programmieraufgabe 1: Paketvermittlung

Ausgabe: Mi. 11.11.2020; Abgabe: Di. 1.12.2020

1 Allgemeine Hinweise zur Bearbeitung

Die Programmieraufgaben sollen in Gruppen mit **maximal fünf Studierenden** bearbeitet werden. Die Implementierung soll in **Java 11** geschehen. Machen Sie sich also zunächst mit der Programmierung in Java (siehe z. B. offizielle Dokumentation der Java API) vertraut.

1.1 Bibliotheken

Zur Lösung der Programmieraufgaben soll ausschließlich auf die durch die Java-Standardbibliothek zur Verfügung gestellten Klassen zurückgegriffen werden, die sich in den folgenden Paketen (oder Subpaketen davon) befinden:

- `java.io.*`
- `java.lang.*`
- `java.net.*`
- `java.nio.*`
- `java.security.*`
- `java.util.*`
- `java.text.*`

Klassen aus den Pakten `com.sun.net.*` bzw. `sun.net.*` oder Bibliotheken von Drittanbietern dürfen nicht eingebunden werden. Ebenfalls ist das Kopieren von Quellcode aus Bibliotheken oder anderen Fremdquellen nicht erlaubt. **Verstöße führen zum Nichtbestehen der Studienleistung!**

1.2 Fehlerbehandlung

Im Programm sollten Exceptions (bzw. Fehlerzustände), die beispielsweise bei arithmetischen Operationen (Division durch 0) oder bei Zugriff auf noch nicht erzeugte Objekte entstehen können, verarbeitet werden. Eine unzureichende Fehlerbehandlung führt zu Punktabzug.

1.3 Kommentare und Dokumentation

Kommentieren Sie Ihren Quelltext ordentlich! Für Abgaben ohne ausreichende Kommentare werden Punkte abgezogen. Zusätzlich sollten Sie alle über die Aufgabenstellung hinausgehenden Bedienungsaspekte der Applikation (Befehle, Kommandozeilenparameter, ...) dokumentieren, um eine reibungslose Korrektur zu ermöglichen. Neben einer Textdatei ist es auch zulässig, diese Dokumentation von der Anwendung selbst (z. B. direkt nach dem Starten) ausgeben zu lassen.

1.4 Abgabe

Die Abgabe findet im Moodle statt. Die Abgabe kann bis zum Abgabetermin beliebig oft geändert werden, wobei die zuletzt abgegebene Version bewertet wird.

Die Abgabe muss vom Compiler übersetzbar sein und unter **Java 11** laufen. Abzugeben ist ein zip-Archiv mit allen zum Projekt gehörenden Quelltexten und eine ausführbare jar-Datei.

2 Tipps zur Programmierumgebung

2.1 IDE

Ein hilfreiches Werkzeug bei der Entwicklung von Software ist eine *integrierte Entwicklungsumgebung* (IDE). Bekannte Java-IDEs sind:

- **IntelliJ IDEA**
- **Eclipse**
- NetBeans IDE
- Editoren mit Java Plugin wie
  - Visual Studio Code
  - Atom
  - viele weitere

Alle genannten IDEs verfügen über detaillierte Anleitungen zur Installation und dem Import von Projekten.

2.2 Git Repositories

Git ist eine freie Software zur verteilten Versionsverwaltung von Dateien, die die gemeinsame Arbeit an den Programmieraufgaben unterstützen kann. Eine kurze Anleitung zu Git finden Sie z. B. hier. Einige Remote Git Server sind:

- (Fakultät/IRB) Gitea

- (Fachschaft) GitLab
- (Microsoft) GitHub
- (Atlassian) Bitbucket

Wichtig: Erstellen Sie *private* Repositories um Plagiate zu vermeiden! Öffentliche Repositories werden von Internet-Suchmaschinen (z. B. Google) indexiert und können somit von anderen Gruppen gefunden werden.

3 Aufgabe

Um Nachrichten in einem paketvermittelten Rechnernetz (z. B. dem Internet) von einem Sender End-System zu einem Empfänger End-System übertragen zu können, müssen längere Nachrichten in einzelne Datenpakete aufgeteilt werden. Die Datenpakete können anschließend entweder verbindungsorientiert oder verbindungslos vom Sender End-System zum Empfänger End-System übertragen werden.

Ein Datenpaket soll in dieser Aufgabe aus den folgenden Feldern bestehen:

- **Version:** 4 oder 6
- **Absender:** je nach Version IPv4- oder IPv6-Adresse
- **Empfänger:** je nach Version IPv4- oder IPv6-Adresse
- **Paketlaufnummer:** Integer > 0
- **Datenteil-Länge:** Integer > 0
- **Datenteil:** String aus Buchstaben, Zahlen, Sonderzeichen

Ihre Aufgabe besteht nun darin ein Java-Programm zu entwickeln, dass vom Benutzer die Version, den Absender, den Empfänger und eine mehrzeilige Nachricht der Länge N, die mit `Zeilenumbruch.Zeilenumbruch` (bzw. <CR><LF>.<CR><LF>) abgeschlossen wird, bekommt und die Nachricht in Datenpakete mit Datenteil-Länge P verpackt. Dabei darf jedes Paket ausschließlich ganze Wörter enthalten, eine Trennung in der Wortmitte ist also ausgeschlossen.

Weitere Vorgaben:

- Die maximale Datenteil-Länge P soll als Kommandozeilenargument übergeben werden.
- Die Nachricht enthält keine Leerzeichen am Zeilenanfang und am Zeilenende.
- Keine Anfangs- und Endleerzeichen im Datenteil.
- Leerzeichen haben die Länge 1.

- Zeilenumbrüche sollen mit `\n` kodiert werden und haben demzufolge die Länge 2. Eine Trennung von `\n` ist nicht erlaubt.
- Wenn die Nachricht vom Benutzer ein Wort mit Länge $W > P$ enthält, so gibt dass Java-Programm einen entsprechenden Fehler aus. Beispielhafte Fehlermeldung: *Die Nachricht kann nicht versendet werden, da sie ein Wort mit Länge $W > P$ enthält.* Dabei W und P mit den eigentlichen Werten ersetzen.
- Wörter mit Bindestrich (z. B. Peer-To-Peer-Modell) oder Schrägstrich (z. B. TCP/IP) dürfen beliebig am Bindestrich/Schrägstrich getrennt und in mehrere Datenpakete aufgeteilt werden.
- Es sind nur Datenpakete mit Version 4 oder 6 erlaubt. Die Eingaben für Absender und Empfänger müssen jedoch **nicht** auf Korrektheit geprüft werden.

4 Beispiele zur Paketaufteilung

Es folgen einige Beispiele, die nur die Aufteilung von Nachrichten in Pakete verdeutlichen. Das Command Line Interface (CLI) für das Java-Programm ist in Abschnitt 5 angegeben.

Beispiel 1

```
Maximale Datenteil-Länge: 5
Nachricht:
Hello World!
Aufteilung in 3 Pakete:
["Hello", "World", "!"]
```

Beispiel 2a

```
Maximale Datenteil-Länge: 2
Nachricht:
a b c d e
Aufteilung in 5 Pakete:
["a", "b", "c", "d", "e"]
```

Beispiel 2b

```
Maximale Datenteil-Länge: 3
Nachricht:
a b c d e
Aufteilung in 3 Pakete:
["a b", "c d", "e"]
```

Beispiel 3a

```
Maximale Datenteil-Länge: 4
Nachricht:
Peer-To-Peer
Aufteilung in 3 Pakete:
["Peer", "-To-", "Peer"]
```

Beispiel 3b

```
Maximale Datenteil-Länge: 5
Nachricht:
Peer-To-Peer
Aufteilung in 3 Pakete:
["Peer-", "To-", "Peer"]
```

Beispiel 4

```
Maximale Datenteil-Länge: 7
Nachricht:
Hi Alice,

das ist ein Test.

Viele Grüße

Bob
Aufteilung in 8 Pakete:
["Hi", "Alice,", "\n\ndas", "ist ein", "Test.\n", "\nViele", "Grüße\n", "\nBob"]
```

5 Command Line Interface

Das Java-Programm soll als Kommandozeilenprogramm realisiert werden.

```
Die maximale Datenteil-Länge ist 7. // per Kommandozeilenargument übergeben

Version: 4              // "4" vom Benutzer eingegeben
Absender: 1.2.3.4       // "1.2.3.4" vom Benutzer eingegeben
Empfänger: 127.0.0.0    // "127.0.0.0" vom Benutzer eingegeben
Nachricht:
Hi Alice,               // Es folgt die
                        // mehrzeilige
das ist ein Test.       // Nachricht(eneingabe).
```

```
Viele Grüße

Bob                     // <CR><LF> und
.                       // .<CR><LF> markieren nur das Ende der Eingabe
                        // und gehören nicht mehr zur Nachricht.
Es sind 8 Datenpakete notwendig. // Anzahl der Datenpakete ausgeben. Hier: 8

Version: 4                  // Die Felder des
Absender: 1.2.3.4           // 1. Datenpakets
Empfänger: 127.0.0.0        // ausgeben.
Paketlaufnummer: 1
Datenteil-Länge: 2
Datenteil: Hi

Version: 4                  // Die Felder des
Absender: 1.2.3.4           // 2. Datenpakets
Empfänger: 127.0.0.0        // ausgeben.
Paketlaufnummer: 2
Datenteil-Länge: 6
Datenteil: Alice,

Version: 4                  // Die Felder des
Absender: 1.2.3.4           // 3. Datenpakets
Empfänger: 127.0.0.0        // ausgeben.
Paketlaufnummer: 3
Datenteil-Länge: 7
Datenteil: \n\ndas

Version: 4                  // Die Felder des
Absender: 1.2.3.4           // 4. Datenpakets
Empfänger: 127.0.0.0        // ausgeben.
Paketlaufnummer: 4
Datenteil-Länge: 7
Datenteil: ist ein

Version: 4                  // Die Felder des
Absender: 1.2.3.4           // 5. Datenpakets
Empfänger: 127.0.0.0        // ausgeben.
Paketlaufnummer: 5
Datenteil-Länge: 7
Datenteil: Test.\n

Version: 4                  // Die Felder des
Absender: 1.2.3.4           // 6. Datenpakets
Empfänger: 127.0.0.0        // ausgeben.
Paketlaufnummer: 6
Datenteil-Länge: 7
Datenteil: \nViele
```

```
Version: 4                  // Die Felder des
Absender: 1.2.3.4           // 7. Datenpakets
Empfänger: 127.0.0.0        // ausgeben.
Paketlaufnummer: 7
Datenteil-Länge: 7
Datenteil: Grüße\n

Version: 4                  // Die Felder des
Absender: 1.2.3.4           // 8. Datenpakets
Empfänger: 127.0.0.0        // ausgeben.
Paketlaufnummer: 8
Datenteil-Länge: 5
Datenteil: \nBob
```